Yassine Errochdi

Rapport Shavadoop

Table des matières

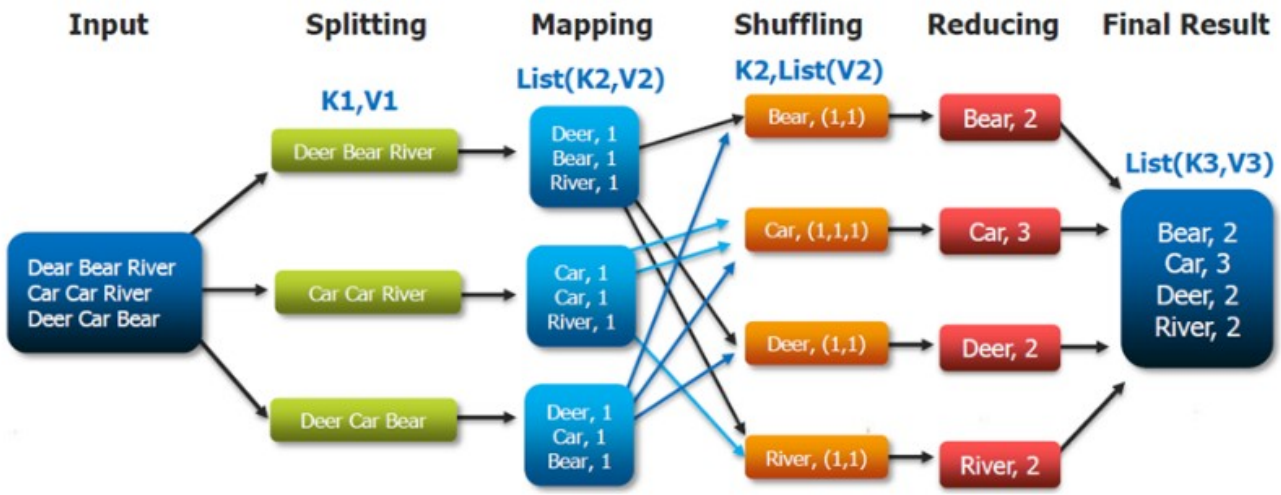
1 Introduction

Ce document a pour objectif la description du travail effectué lors du projet Shavadoop.

L'objectif du projet est d'implémenter une fonction de « WordsCount » on utilisant le mécanisme « Mapreduce ».

L'objectif de l'application est de compter le nombre de mot contenus dans un fichier texte.

La figure ci dessous schématise les différentes étapes du « map Reduce » :



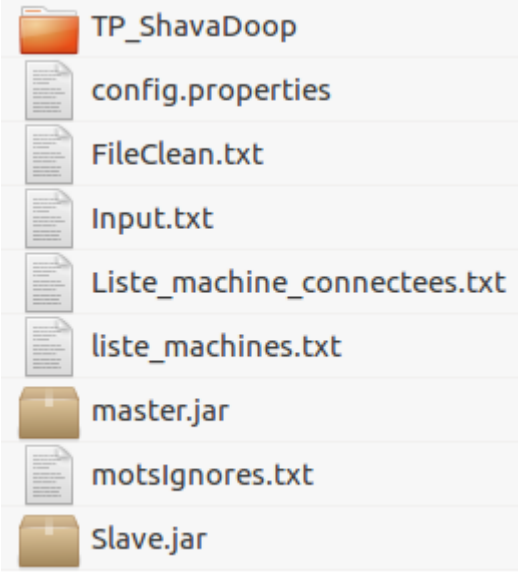
2 Livraison

Ci dessous les différents éléments de livraison :

Le répertoire « sources » contient les deux projets :

- Master : Découpage du fichier d'entrée « Input.txt », et lancement des différentes slave.
- Slave : l'exécution des étapes du MapReduce.

Le répertoire « Appliction_JAR » :



- Slave.jar et master.jar : les exécutables du master et du slave.
- Input.txt : Le fichier texte contenant les mots à compter.
- FileClean.txt : Le fichier texte nettoyer (suppression des mots non important)
- motIgnores.txt : La liste des mots à supprimer du ficher d'entrée.
- liste_machines.txt: Liste des machines utilisées.
- liste_machine_conectees.txt : Liste des machines accessibles.
- Config.properties : Le fichier contenant les variables utilisées par l'application. Le fichier est a modifier afin d'adapter l'application au contexte

utilisateur.

- TP_ShavaDoop : Répertoire dont le contenu est recréé à chaque lancement.Il contient des répertoires contenant les différents fichiers générés selon l'avancement.

  Le fichier final sera crée dans le répertoire « MAP »

3 Exécution

3.1 Préparation

Avant le lancement de l'application l'utilisateur doit paramétrer le fichier « config.properties ».

```
CheminReps=/cal/homes/errochdi/shava/TP_ShavaDoop
CheminJar=/cal/homes/errochdi/shava
userName=errochdi
inputFilename=/cal/homes/errochdi/shava/Input.txt
hostsFilename=/cal/homes/errochdi/shava/liste_machines.txt
```

Comme indiqué ci-dessus le fichier « config.properties » sert à paramétrer les variables utilisée par l'application ci-dessous une description :

- CheminRep : l'endroit où les fichiers de sortie seront crées.
- CheminJar : l'endroit contenant les fichiers JAR.
- UserName : Le nom de l'utilisateur (utilisé dans le lancement des JAR slaves).
- InputFilename : Le fichier d'entrée (Input.txt)
- hostsFilename : Le fichier contenant l'adresse des différentes machines utilisées.

3.2 Lancement

L'exécution se fait en lançant le master.jar avec le chemin du fichier de configuration en paramètre d'entrée.

```
$ java -jar master.jar /Livraison_Shavadoop/Application_JAR/config.properties
```

4 Fonctionnement

4.1 Fonctionnement du Master

Le Main du master lancera une à une les différentes étapes du « Mapreduce » , chaque étape est chronométrée:

1) Nettoyage du répertoire de sortie.

   Tous les dossiers et fichiers du répertoire de sortie sont supprimés.

2) Instanciation de la classe master.

   Outre l'instanciation de classe, une méthode est lancée afin de vérifier la connexion aux postes.

   Une fois les connexions vérifiées un fichier est crée avec la liste des machines connectées.

3) Nettoyage du fichier input :

   Le nettoyage du fichier input.txt enlèvera tous les mots non comptabilisés (motIgnores.txt).

4) Splitting : Création des fichiers SX (1 fichier par ligne).

5) Mapping : Lancement des slaves afin de créer les fichiers Umx.

   Afin de bien répartir le calcul entre les différentes machines on calcule un modulo entre le numéro du SX et le nombre de machines.

   Si le nombre de ligne est inférieur au nombre de machines on prend la machine correspondante à la ligne traitée sinon on prend le modulo

6) Création des dictionnaires : Deux dictionnaire sont crées "UMx – machines" et "clés -Umx".

7) Shuffling : Le Shuffling se basera sur les dictionnaires crées afin de regrouper les mots dans un seul fichier SMX puis un fichier Rmx avec une seule ligne <Clé,valeur>. Le regroupement se fera par le lancement des

slaves avec l'option : modeUMXSMX.

8) Reducing : fusionner les différents fichiers RMX et création du fichier final « map_reduce.txt »

4.2 Fonctionnement du Slave

Le slave lance les fonctions du mapping et reducing sur les différentes machines.

Le slave gérera modes :

- Mode SX → Umx
- Mode Umx → Smx → Rmx

5 Problèmes

Un des problème majeur que j'ai rencontré lors de ce projet, et la gestion des threads.

En effet, pour cause du nombre trop important de threads créés, les threads ne se terminaient pas correctement ce qui cause une perte d'information.

La solution envisagée est de mettre en place un mécanisme de gestion de « timeout », ainsi si l'exécution du thread dépasse un certain temps (définit dans le code), le thread est stocké dans une liste afin qu'il soit identifié, et traité dans un deuxième temps.

6 Résultats

6.1 Fichier : code forestier Mayotte

Mot	Nombre d'occurrences
biens	8
forestier	7
code	6
partie	5
gestion	4
dispositions	4
agroforestie	4

Mot	Nombre d'occurrences
rs	
communes	4
mayotte	4
forestiers	4
legislative	3
publiques	2
preliminaire	2
agroforestiere	2
souscrivent	2
volontairement	2
proprietaires	2
livre	2
titre	2
attachees	2
demembrer	2
regis	2
bonne	2
presentant	2
prioritairement	2
unite	2
l021	2
ii	2
accorde	2
forestiere	2
table	2
propriete	2

Mot	Nombre d'occurrences
fait	2
aides	2
article	2
constitue	2
garanties	2
benefice	2
engagement	2
plan	1
matieres	1
sommaire	1